```
/**
 To inject HTML into a page you can use .innerHTML to inject it into your .html file.
 Place the HTML into a variable then use .innerHTML to inject it. Use `` backticks to
 encapsulate the HTML. Don't forget the querySelector(); to pull in the class or id.
 it goes like this;
 const div = querySelector(.class);
 div.innerHTML = <variable holding HTML>

 Until the the HTML is dumped into the DOM it is a string. So you can not add classes,
eventListeners or other things to it. So if you wish to change any of the attributes you need
to first dump it into the DOM. Then from there once it is in the DOM you can then add
whatever attributes you want.
 */

const item = document.querySelector('.item');
const src = `https://picsum.photos/200`;
const desc = `Cute Pup`;

const myHTML = `
<div class="wrapper">
<h2>${desc}<h2>
<img src="${src}" alt="${desc}"/>
<div>
`;

item.innerHTML = myHTML;

console.log(myHTML);

/**
 If you want to create HTML from a string there are several ways to do it, here are two ways.
 This is one way you can add a class after .innerHTML injection. Most of the time you will
use this way.
 */
const itemImage = document.querySelector('.wrapper img');
itemImage.classList.add('round');
console.log(itemImage);

/*OR you can convert a string into a DOM element from within Javascript
using .creatRange().createContextualFragment(pass in the variable name), so now you can add
classes, eventListeners and other things to them before you insert them into the DOM. See
below.
 */

const myFragment = document.createRange().createContextualFragment(myHTML);
console.log(myFragment);

/**
 * Now you can querySelector() elements from myFragment.
 */
const myImage = myFragment.querySelector('img');
console.log(myImage);

/**
 append myFragment to the body.

 */

document.body.appendChild(myFragment);

/**
 * There are xss security issues with this. In the security video will cover how to avoid
```

```
this.
 */
```